Лабораторная работа №2
«Деревья решений»

Цель работы: получить практические навыки работы с методом деревьев решений на практических примерах с использованием языка программирования python.

Теоретический материал
Основные понятия

Дерево решений – метод автоматического анализа данных для построения классификационных и регрессионных моделей, является как методом извлечения, так и одновременно методом представления данных. Дерево решений представляет способ представления правил в иерархической, последовательной структуре, где каждому объекту соответствует единственный узел, дающий решение. Под правилом понимается логическая конструкция, представленная в виде «если …, то …».

Рисунок 1 Классификация с помощью дерева решений

Таким образом, у дерева решений есть два вида отображения:

1. в виде графической структуры «дерево»
2. в виде списка правил «если …, то …»

Деревья решений отлично справляются с задачами классификации, т.е. отнесения объектов к одному из ранее известных классов. Целевая переменная должна иметь дискретные значения.

Алгоритмы построения дерева решений

Рассмотрим два основных метода, которые используют деревья принятие решений.

CART (sklearn.tree.DecisionTreeClassifier) был первым из методов, придуманный в 1983 четверкой известных ученых в области анализа данных: Leo Breiman, Jerome Friedman, Richard Olshen and Stone.

Суть этого алгоритма состоит в обычном построении дерева принятия решений. На первой итерации строятся все возможные (в дискретном смысле) гиперплоскости, которые разбивают пространство на два. Для каждого такого разбиения пространства считается количество наблюдений в каждом из подпространств разных классов. В результате выбирается такое разбиение, которое максимально выделило в одном из подпространств наблюдения одного из классов. Соответственно, это разбиение будет нашим корнем дерева принятия решений, а листами на данной итерации будет два разбиения.

На следующих итерациях мы берем один худший (в смысле отношения количества наблюдений разных классов) лист и проводим ту же операцию по разбиению его. В результате этот лист становится узлом с каким-то разбиением, и двумя листами.

Продолжаем так делать, пока не достигнем ограничения по количеству узлов, либо от одной итерации к другой перестанет улучшать общая ошибка (количество неправильно классифицированных наблюдений всем деревом). Однако, полученное дерево будет «переобучено» (будет подогнано под обучающую выборку) и, соответственно не будет давать нормальные результаты на других данных. Для того, чтобы избежать «переобучения», используют тестовые выборки (либо кросс-валидацию) и, соответственно, проводится обратный анализ (так называемый pruning), когда дерево уменьшают в зависимости от результата на тестовой выборке.

Это относительно простой алгоритм, в результате которого получается одно дерево принятия решений. За счет этого, он удобен для первичного анализа данных.

+ Быстрое построение модели

+ Легко интерпретируется (из-за простоты модели можно легко отобразить дерево и проследить за всеми узлами дерева)

– Часто сходится на локальном решении (к примеру, на первом шаге была выбрана гиперплоскость, которая максимально делит пространство на этом шаге, но при этом не приведет к оптимальному решению)

Random Forest (sklearn.ensemble.RandomForestClassifier) – метод, придуманный после CART одним из четверки – Leo Breiman в соавторстве с Adele Cutler, в основе которого лежит использование ансамбля деревьев принятия решений.

Суть алгоритма в том, что на каждой итерации делается случайная выборка переменных, после чего на этой новой выборке запускают построение дерева принятия решений. При этом производится «bagging» - выборка случайных двух третей наблюдений для обучения, а оставшаяся треть используется для оценки результата. Такую операцию проделывают сотни или тысячи раз. Результирующая модель будет результатом «голосования» набора полученных при моделировании деревьев.

+ Высокое качество результата, особенно для данных с большим количеством переменных и малым количеством наблюдений

+ Возможность распараллеливания

+ Не требуется тестовая выборка

– Каждое из деревьев огромное, следовательно, в результате модель получается огромная

– Долгое построение модели для достижения хороших результатов

– Сложная интерпретация модели (сотни или тысячи больших деревьев сложны для интерпретации)

Ход работы

1. Прочитать теоретическую часть по деревьям решений
2. Описать структуру исходных данных для своего набора:
   a. общие характеристики массива данных: предметная область, количество записей
   b. входные параметры: названия и типы
   c. выходной класс: название и значения
3. Провести серию экспериментов с построением и тестированием деревьев решений (используя DecisionTreeClassifier и RandomForestClassifier), переразбивая исходное множество данных, заданное в варианте, следующим образом:

Номер эксперимента	Размер обучающей выборки	Размер тестовой выборки
1	60 %	40 %
2	70 %	30 %
3	80 %	20 %
4	90 %	10 %

4. Осуществить классификацию
5. Сформулировать вывод по использованию деревьев решений для исходной задачи

Пример выполнения доступен по адресу:
https://github.com/artemvirused/ITMO_ML_Labs/blob/master/Lab2/ML_lab2.ipynb